运输层

运输层之间的通信是进程与进程之间的，通过端口的

一、运输层协议概述

1、从通信和信息处理的角度看，运输层向它上面的应用层提供通信服务，它属于面向通信部分的最高层，同时也是用户功能中的最低层。

2、当网络的边缘部分中的两个主机使用网络的核心部分的功能进行端到端的通信时，只有位于网络边缘部分的主机的协议栈才有运输层，　　而网络核心部分中的路由器在转发分组时都只用到下三层的功能。

3、两个主机进行通信实际上就是两个主机中的应用进程互相通信。

4、运输层的一个很重要的功能就是复用和分用。通过端口实现。

5、网络层为主机之间提供逻辑通信，运输层为应用进程之间提供端到端的逻辑通信

6、当运输层采用面向连接的　　TCP 协议时，尽管下面的网络是不可靠的　　（只提供尽最大努力服务），但这种逻辑通信信道就相当于一条全双工的可靠信道。当运输层采用无连接的　　UDP 协议时，这种逻辑通信信道是一条不可靠信道。

7、两个对等运输实体在通信时传送的数据单位叫作运输协议数据单元　　TPDU

8、 UDP 在传送数据之前不需要先建立连接。虽然　　UDP 不提供可靠交付，但在某些情况下 UDP 是一种最有效的工作方式；　　TCP 则提供面向连接的服务。

9、运输层的　UDP 用户数据报与网际层的　　IP 数据报的区别：　IP 数据报要经过互连网中许多路由器的存储转发，但　　UDP 用户数据报是在运输层的端到端抽象的逻辑信道中传送的。

10、硬件端口与软件端口的区别　　：在协议栈层间的抽象的协议端口是软件端口。　　路由器或交换机上的端口是硬件端口。　硬件端口是不同硬件设备进行交互的接口，　　而软件端口是应用层的各种协议进程与运输实体进行层间交互的一种地址。

11、端口用一个　16 位端口号进行标志。

12、端口号只具有本地意义，　　即端口号只是为了标志本计算机应用层中的各进程。　　在因特网中不同计算机的相同端口号是没有联系的。

13、端口的分类：熟知端口和登记端口号合称为服务器端使用的端口号

熟知端口，数值一般为　0~1023。

登记端口号，数值为　1024~49151，为没有熟知端口号的应用程序使用的。使用这个范围的端口号必须在　IANA　登记，以防止重复。

客户端口号或短暂端口号，数值为　　49152~65535，留给客户进程选择暂时使用。

二、用户数据报协议　UDP

1、 UDP 只是在 IP 的数据报服务之上增加了端口的功能和差错检测的功能。

2、 UDP 的主要特点

UDP 是无连接的，即发送数据之前不需要建立连接。

UDP 使用尽最大努力交付，即不保证可靠交付，同时也不使用拥塞控制。

UDP 是面向报文的。

UDP 支持一对一、一对多、多对一和多对多的交互通信。

UDP 的首部开销小，只有　8 个字节。

UDP 没有拥塞控制，很适合多媒体通信的要求

3、发送方　UDP 对应用程序交下来的报文，在添加首部后就向下交付　　IP 层。 UDP 对应用层交下来的报文，既不合并，也不拆分，而是保留这些报文的边界。

4、 UDP 的首部格式（每个部分各两个字节）

源端口（不需要回送时可填　　0），目的端口，长度（包括数据部分，最小为　　8 字节），检验和

（检查用户数据报是否有差错）

5、 IP 数据报首部检验和与 UDP 数据报检验和的区别： IP 数据报只检验首部， UDP 则把首部和数据报部分一起检验了。

6、伪首部与检验和：为了计算检验和在 UDP 首部之前再加入 12 字节的伪首部。伪首部不能向上递交也不用向下传送。接收方将 UDP 数据报与伪首部按二进制反码求和，正确的结果应为全 0，否则丢弃这个数据报。

7、伪首部包括源 IP，目的 IP，一个全 0 字节，一个协议字段， 2 字节的 UDP 长度

三、传输控制协议 TCP

1、 TCP 最主要的特点

TCP 是面向连接的运输层协议。

每一条 TCP 连接只能有两个端点，每一条 TCP 连接只能是点对点的（一对一） 。

TCP 提供可靠交付的服务。

TCP 提供全双工通信。

面向字节流。

2、 TCP 连接是一条虚连接而不是一条真正的物理连接。

3、 TCP 连接的端点：不是主机，不是主机的 IP 地址，不是应用进程，也不是运输层的协议端口，而是套接字 (socket)或插口，是端口号拼接到 IP 地址后面构成的。

套接字 socket = (IP 地址 : 端口号) 192.168.1.1 ： 80

TCP 连接 ::= {socket1, socket2}

= {(IP1: port1), (IP2: port2)}

4、同一个 IP 地址可以有多个不同的 TCP 连接，而同一个端口号也可以出现在多个不同的 TCP 连接中。

四、可靠传输的工作原理

1、理想传输条件的两个特点：传输信道不产生差错，无论发送发以多快的速度发送，接送方总来的及接受数据。在这种条件下不需要采取何种措施就可以实现可靠传输。

2、停止等待协议：每发送完一个分组就停止发送，等待对方确认。确认后再发下一个分组

3、为了在出现差错时能够继续通信：在发送完一个分组后，必须暂时保留已发送的分组的副本； 分组和确认分组都必须进行编号； 超时重传， 超时计时器的重传时间应当比数据在分组传输的平均往返时间更长一些。

4、确认丢失和确认迟到

5、使用确认和重传机制，我们就可以在不可靠的传输网络上实现可靠的通信。

6、可靠传输协议又称为自动重传请求 ARQ

7、信道利用率： $U = \frac{T_D}{T_D + RTT + T_A}$ TD ：分组发送时间； RTT ：分组往返时间； TA ：发送确认分组所需的时间

8、停止等待协议的优点是简单，但缺点是信道利用率太低。

9、提高信道利用率的措施：采用流水线传输：发送方可连续发送多个分组，不必每发完一个分组就停顿下来等待对方的确认

10、连续 ARQ 协议： TCP 连接的每一端都必须设有两个窗口 —— 一个发送窗口和一个接收窗口。 发送窗口内的数据可以连续发送出去， 不需要等待对方的确认， 接收方采用累积确认，对按序到达的最后一个分组确认，表示：到这个分组为止的所有分组都已正确收到了。

发送方每接收到一个确认，就把发送窗口向前滑动一个分组单位。

五、 TCP 报文段的首部格式

1、源端口和目的端口字段 —— 各占 2 字节。端口是运输层与应用层的服务接口。运输层的复用和分用功能都要通过端口才能实现。

2、序号字段 —— 占 4 字节。TCP 连接中传送的数据流中的每一个字节都编上一个序号。 序号字段的值则指的是本报文段所发送的数据的第一个字节的序号。

3、确认号字段 —— 占 4 字节，是期望收到对方的下一个报文段的数据的第一个字节的序号。

4、数据偏移（即首部长度） —— 占 4 位，它指出 TCP 报文段的数据起始处距离 TCP 报文段的起始处有多远，以四个字节为单位，其实就是首部长度，最大 60 字节。

5、确认 ACK —— 只有当 ACK =1 时确认号字段才有效。

6、复位 RST (ReSeT) —— 当 RST =1 时，表明 TCP 连接中出现严重差错（如由于主机崩溃或其他原因），必须释放连接，然后再重新建立运输连接。

7、同步 SYN —— 同步 SYN = 1 表示这是一个连接请求或连接接受报文

8、终止 FIN (FINis) —— 用来释放一个连接。 FIN =1 表明此报文段的发送端的数据已发送完毕，并要求释放运输连接。

9、窗口字段 —— 占 2 字节，用来让对方设置发送窗口的依据，单位为字节。

10、检验和 —— 占 2 字节。检验和字段检验的范围包括首部和数据这两部分。检验规则和 UDP 检验和一样。

11、紧急指针字段 —— 占 16 位，指出在本报文段中紧急数据共有多少个字节

12、 MSS 是 TCP 报文段中的数据字段的最大长度

13、选项字段 —— 长度可变。包括：

窗口扩大选项 —— 占 3 字节，

时间戳选项 —— 占 10 字节，其中最主要的字段时间戳值字段（ 4 字节）和时间戳回送回答字段（ 4 字节）。用来计算往返时间，处理 TCP 序号超过 2 的 32 次方的情况，防止序号绕回，时间戳可以做个标记

选择确认选项

14、填充字段，填 0，这是为了使整个首部长度是 4 字节的整数倍。

六、 TCP 可靠传输的实现

1、 TCP 的滑动窗口以字节为单位

2、窗口越大，发送方就可以在收到确认之前发送更多的数据，传输效率高，但是接收方必须来的及接收。

3、 窗口的分类：发送窗口，可用窗口，未确认窗口

4、 A 的发送窗口并不总是和 B 的接收窗口一样大（因为有一定的时间滞后） 。

5、 TCP 要求接收方必须有累积确认的功能，这样可以减小传输开销。

6、缓存和窗口的关系：缓存一般大于窗口

7、发送缓存与接收缓存的作用

发送缓存用来暂时存放：

发送应用程序传送给发送方 TCP 准备发送的数据；

TCP 已发送出但尚未收到确认的数据。

接收缓存用来暂时存放：

按序到达的、但尚未被接收应用程序读取的数据；

不按序到达的数据。

8、超时重传时间的选择： $RTO = RTT_S + 4 \times RTT_D$

平均往返时间

新的 $RTT_S = (1 - \alpha) \times$ (旧的 RTT_S) $+ \alpha \times$ (新的 RTT 样本)

$0 \leq \alpha < 1$ 推荐 =0.125

RTT 的偏差的加权平均值

新的 RTT_D = (1 −β) × (旧的 RTT_D) + β× | RTT_S −新的 RTT 样本 |

β推荐 =0.25

9、如何判定一个确认报文段是对原来的报文段 1 的确认，还是对重传的报文段 2 的确认？采用 Karn 算法 :在计算平均往返时间 RTT 时，只要报文段重传了，就不采用其往返时间样本。

修正的 Karn 算法 :报文段每重传一次，就把 RTO 增大一些：

10、选择确认 sack 是解决：接收方收到了和前面的字节流不连续的两个字节块，怎样让发送方只发送没有到达的数据段

11、使用 SACK 时要将 TCP 首部中的 SACK 位置 1，然后再选项字段填入未接受到的断块的边界值， 每个边界值占 4 个字节， 确定一个断块要用两个边界值， 所以选项字段最多只能指明 4 个字节块的信息。另外还要一个选项指明是 SACK 选项，一个指明 SACK 占用的字节数

七、 TCP 的流量控制：利用滑动窗口实现流量控制

1、流量控制 (flow control) 就是让发送方的发送速率不要太快，既要让接收方来得及接收，也不要使网络发生拥塞。

2、丢失报文导致互等死锁的解决办法：持续计时器

TCP 为每一个连接设有一个持续计时器。

只要 TCP 连接的一方收到对方的零窗口通知，就启动持续计时器。

若持续计时器设置的时间到期，就发送一个零窗口探测报文段（仅携带 1 字节的数据），而对方就在确认这个探测报文段时给出了现在的窗口值。

若窗口仍然是零，则收到这个报文段的一方就重新设置持续计时器。

若窗口不是零，则死锁的僵局就可以打破了。

3、发送方的发送窗口不能超过接收方给的接收窗口的数值， TCP 窗口的单位是字节。

4、考虑到传输效率，要解决 TCP 报文段的发送时机的选择 :

长度限制：第一种机制是 TCP 维持一个变量，它等于最大报文段长度 MSS。只要缓存中存放的数据达到 MSS 字节时，就组装成一个 TCP 报文段发送出去。

自己要求： 第二种机制是由发送方的应用进程指明要求发送报文段， 即 TCP 支持的推送 (push)操作。

时间限制：第三种机制是发送方的一个计时器期限到了，这时就把当前已有的缓存数据装入报文段（但长度不能超过 MSS）发送出去。

八、 TCP 的拥塞控制

1、在某段时间，若对网络中某资源的需求超过了该资源所能提供的可用部分，网络的性能就要变坏 —— 产生拥塞，即资源需求 >可用资源

2、拥塞控制与流量控制的关系：

拥塞控制是一个全局性的过程， 就是防止过多的数据注入到网络中， 这样可以使网络中的路由器或链路不致过载。

流量控制往往指在给定的发送端和接收端之间的点对点通信量的控制。 流量控制所要做的就是抑制发送端发送数据的速率，以便使接收端来得及接收。

3、拥塞控制要解决好网络吞吐量与负载的关系

4、几种拥塞控制方法：慢开始和拥塞避免，快重传和快恢复

5、发送方维持一个叫做拥塞窗口 cwnd 的状态变量。拥塞窗口的大小取决于网络的拥塞程度，并且动态地在变化。发送方让自己的发送窗口等于或小于拥塞窗口。

6、发送方控制拥塞窗口的原则是：只要网络没有出现拥塞，拥塞窗口就再增大一些，发送

更多的分组。但只要网络出现拥塞，拥塞窗口就减小一些，以减少注入到网络中的分组数。

7、“拥塞避免”不能完全避免拥塞，只是说在拥塞避免阶段把拥塞窗口控制为较多按线性规律增长，使网络比较不容易出现拥塞。

8、快重传算法：快重传算法首先要求接收方每收到一个失序的报文段后就立即发出重复确认。这样做可以让发送方及早知道有报文段没有到达接收方。发送方只要一连收到三个重复确认就应当立即重传对方尚未收到的报文段。

9、快恢复算法：当发送端收到连续三个重复的确认时，就执行“乘法减小”算法，把慢开始门限 ssthresh 减半，然后执行加法增大。这是因为三个确认报文能够到达发送端，网络很可能没有拥塞

10、发送窗口的上限值 =Min [rwnd, cwnd] 接收方窗口 rwnd 和拥塞窗口 cwnd

九、 TCP 的运输连接管理

1、运输连接就有三个阶段，即：连接建立、数据传送和连接释放。

2、 TCP 连接的建立都是采用客户服务器方式。主动发起连接建立的应用进程叫做客户(client)。被动等待连接建立的应用进程叫做服务器(server)。

3、 TCP 用三次握手建立连接： **A** 表示发送方， **B** 表示接收方

（1）A 的 TCP 向 B 发出连接请求报文段，其首部中的同步位 SYN = 1，并选择序号 seq = x，表明传送数据时的第一个数据字节的序号是 x。

（2）B 的 TCP 收到连接请求报文段后，如同意，则发回确认。B 在确认报文段中应使 SYN = 1，使 ACK = 1，其确认号 ack = x +1，自己选择的序号 seq = y。

（3）A 收到此报文段后向 B 给出确认，其 ACK = 1，确认号 ack = y +1。 A 的 TCP 通知上层应用进程，连接已经建立。

4、 TCP 的连接释放：四次握手释放连接。 **A** 表示发送方， **B** 表示接收方

（1）A 把连接释放报文段首部的 FIN = 1，其序号 seq = u，等待 B 的确认。

（2）B 发出确认，确认号 ack = u +1，而这个报文段自己的序号 seq = v。 TCP 服务器进程通知高层应用进程。 从 A 到 B 这个方向的连接就释放了， TCP 连接处于半关闭状态。B 若发送数据， A 仍要接收。

（3）若 B 已经没有要向 A 发送的数据，其应用进程就通知 TCP 释放连接。

（4）A 收到连接释放报文段后，必须发出确认。

5、数据传输结束后，通信的双方都可释放连接。

6、发送方确认后必须等待 2MSL 的时间后才能真正释放连接理由如下：

第一，为了保证 A 发送的最后一个 ACK 报文段能够到达 B。

第二，防止“已失效的连接请求报文段”出现在本连接中。 A 在发送完最后一个 ACK 报文段后，再经过时间 2MSL，就可以使本连接持续的时间内所产生的所有报文段，都从网络中消失。

整理者：福州大学张毅